Методология БЭМ: что это такое, зачем нужно и обязательно ли применять

Методология БЭМ – это вариант записи названия классов в HTML-коде, разработанный в Яндексе. Расшифровывается как “блок, элемент, модификатор”. Такой стиль записи названия классов позволяет более точно выстраивать иерархию внутри документа, не путаться в названии и функционале элементов при задании им стилей в CSS или модели поведения в JS-скрипте. Еще БЭМ помогает придумывать неограниченное количество классов, которые не будут повторяться между собой. Это очень актуально для больших документов, где есть много практически идентичных блоков, которым, тем не менее, нужно прописывать уникальные стили.

Изначально БЭМ создавался для внутренних потребностей Яндекса и не был объединен в общую методологию. **Постепенно команды разработчиков решили стандартизировать правила и сделать их стандартом для компании.** Теперь же БЭМом пользуется большинство разработчиков из России, СНГ и даже есть из дальнего зарубежья.

Какие задачи решает использование БЭМ-методологии

Разработчики из Яндекса при создании методологии собирались решить такие проблемы:

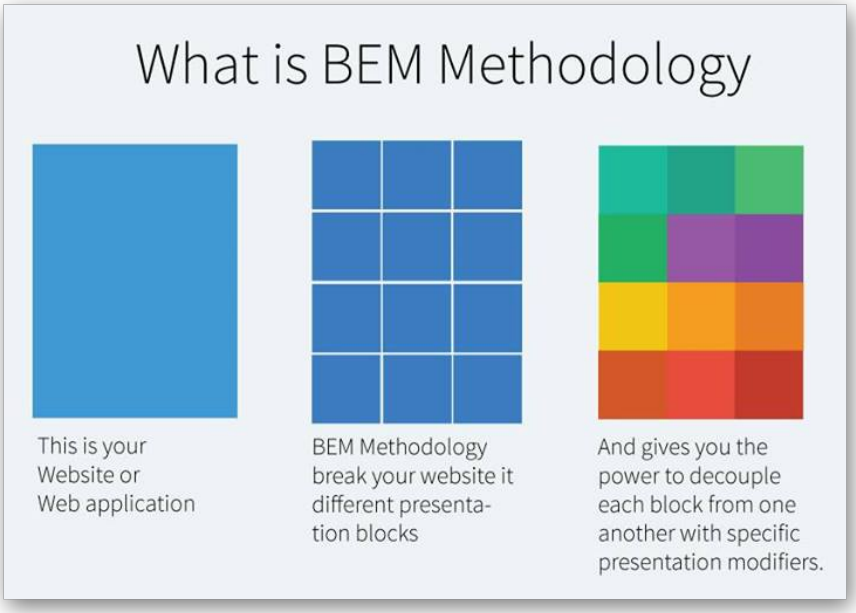
1. Понимаемость кода как для самого разработчика, так и для других разработчиков. Дело в том, что отсутствие общепринятой структуры создает путаницу и усложняет рабочий процесс. Разработчик, знакомый с БЭМ, сможет достаточно быстро самостоятельно разобраться как в своем коде даже по прошествии нескольких лет, так и в коде коллег.
2. Возможность использования любого блока кода повторно без необходимости писать все с нуля. В случае с БЭМ достаточно только заранее создать базу с классами и идентификаторами, которые будут применяться в качестве готовых наработок по мере написания кода.
3. Коммуникация между разными специалистами проекта: дизайнерами, менеджерами, разработчиками. Все они должны называть вещи примерно одними именами для более эффективной работы.
4. Быстрая интеграция специалистов из разных команд. Когда во всех командах написание кода происходит по универсальной методологии процесс интеграции происходит быстрее.
5. Порог входа при переходе с проекта на проект сильно снижается при использовании универсальных методологий.

БЭМ пригодится не только компания и командам разработчиков, но и фрилансерам. Благодаря тому, что методология входит в профессиональную среду, качество и скорость работы над проектами увеличиваются. Освоить принципы работы БЭМ без проблем может даже начинающий разработчик – по-сути, это правила названий элементов в верстке.

Основные правила верстки

Ранее большие проекты представляли собой набор статических HTML-страниц с подключенным к ним файлами со стилями и скриптами. Если в проект требовалось вносить какое-то изменений, то приходилось “перелопачивать” все файлы и вносить их вручную. С

использованием методологий вроде БЭМ процесс значительно упростился – достаточно только заменить название класса или сделать к нему необходимую приписку. Также использование БЭМ помогает сделать более структурированными CSS-файлы, без необходимости делать кучу блоков кода для одного и того же элемента.



Краткая иллюстрация сути BEM-методологии

В своей основе БЭМ использует три взаимосвязанных компонента:

- “Блок” – самый главный, в него входят оставшиеся два элемента;
- “Элемент” – какая-то часть блока, например, кнопка или заголовок;
- “Модификатор” – какая-то определенная характеристика конкретного элемента или блока, например, блок с тенью или кнопка с градиентом.

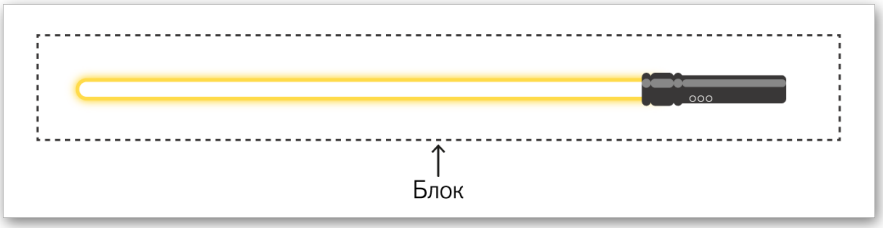
Далее подробно разберем все нюансы работы с каждым из трех компонентов в рамках методологии БЭМ.

Блок

Основной и функционально независимый элемент каждой страницы. Предусматривается, что у него может быть свое поведение, стили, шаблоны, документация и так далее. Вся страница состоит из связанных между собой блоков. Иногда даже в основной блок могут вкладываться другие

блоки, если задачу невозможно корректно реализовать через использование элементов. Они могут использоваться в любом месте страницы, повторяться между собой. Одинаковые блоки даже иногда “кочуют” из проекта в проект.

Блок – это основа любой страницы, поэтому в системе наименований БЭМ наименование блока идет на первом месте в классе. Например, nav – общее название раздела с навигацией.



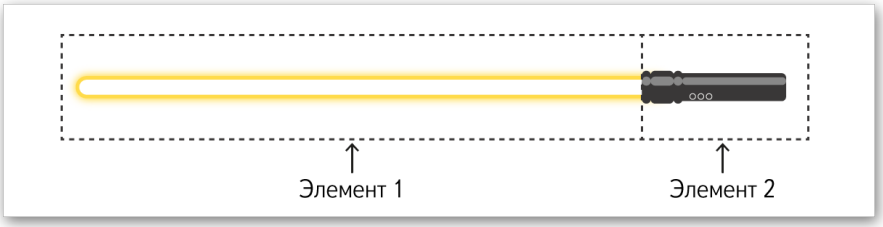
На этой иллюстрации в качестве блока выступает весь световой меч

Элемент

Это уже составная часть блока, поэтому не может использоваться в отрыве от родителя, следовательно, наследует от него часть свойств. В отличие от блоков внутрь элемента не рекомендуется вкладывать другие элементы, хотя технически это возможно. Если в этом возникла необходимость, то лучше тогда обозначить отдельный блок. **Возьмем пример с nav:**

- логотип и телефон можно вынести в качестве элементов, так как внутри них вряд ли предполагается размещение каких-то дополнительных компонентов, кроме картинки и ссылки;
- меню, если оно сложное, то лучше оформить в качестве вложенного блока. Если же это простое перечисление пунктов списком, то можно оформлять в качестве элемента.

Такие рекомендации по работе с элементами обусловлены тем, что если делать вложения, то будет сложнее вносить корректировки. Следовательно, вся суть методологии БЭМ в таком случае теряется.

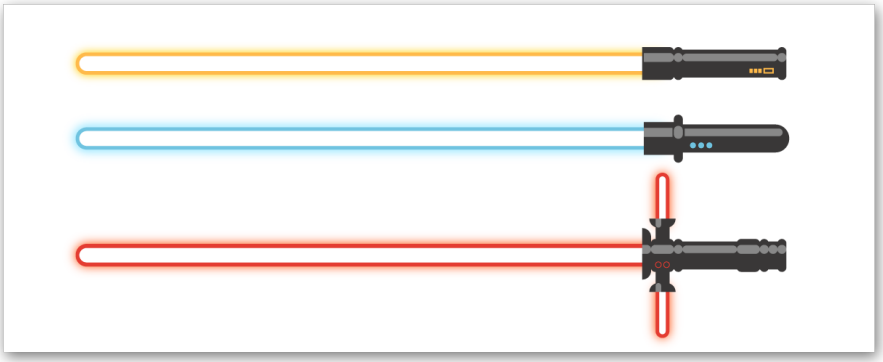


Рукоятка и сам световой мечь это уже элементы

Модификатор

Это уже определенные свойства блока или элемента, например, указание цвета, состояния, поведения. Если элементы не могут быть без блока, а блок не может быть без элементов, то использование модификаторов является опциональным. У модификаторов помимо собственно имени могут быть прописаны и дополнительные значения.

Обычно с помощью модификаторов можно быстро изменить косметические и некоторые функциональные параметры блока/элемента.



Модификаторы это цвет меча, расположение декораций на рукоятки и дополнительный функционал

Правила формирования названий в БЭМ

Задавая имя для компонентов в БЭМ нужно придерживаться следующих правил:

- У каждого БЭМ-компонента должен быть прописан свой класс. Использование идентификаторов нежелательно.
- CSS-свойства могут быть вложенными, но должны прописываться исключительно через классы.

- Разделение слов в названии делается с помощью дефиса (-). Он используется только для разделения двух слов в названии блока, элемента или модификатора, но никак не для отделения разных групп!
- Для отделения от названия блока имени элемента используется двойное подчеркивание (__). Для отделение модификатора одинарное (_).
- Все имена рекомендуется записывать латинскими буквами в нижнем регистре. Цифры ставить можно, но нежелательно.

Вот несколько примеров названий по методологии БЭМ:

- container – название блока;
- container__article – название элемента article, который находится внутри блока container;
- container__article_blue-bg – название модификатора, применяемого к элементу article, находящемуся в блоке container;
- container__button_desabled – булевое название модификатора элемента button в блоке container.

Далее подробно рассмотрим как работать с БЭМ в HTML и CSS.

```
<header class="section-header">
    <div class="section-header-wrapper">
        <div class="herobox">
            <div class="herobox__title">
                <h2>
                    Здравствуйте,
                </h2>
                <h1>
                    Я Имя Фамилия
                </h1>
            </div>
            <div class="herobox__subtitle">
                <h2>
                    front-end разработчик, создам лучший сайт для вас
                </h2>
            </div>
            <div class="herobox__calltoaction">
                <button>Узнать больше</button>
            </div>
        </div>
        <div class="section-header-wrapper__image"></div>
    </div>
</header>
```

Пример рабочего проекта с использованием БЭМ-наименований классов

Запись БЭМ-наименований в HTML

Здесь элементам присваивается класс, название которого составлено в соответствии с методологией БЭМ. **Вот пример куска кода:**

```
<div class="block-name">
  <div class="block-name__elem_red"></div>
  <div class="block-name__elem_black"></div>
</div>
```

В этом простом примере блок кода привязан к одному DOM-узлу, но технически DOM-узел может иметь несколько наименований, что принято называть миксом. **Он используется для:**

- объединения классификации нескольких БЭМ-компонентов без необходимости дублирования кода;
- применения разных вариантов названий, например, обычных и БЭМ-классов;
- выполнять позиционирование вложенных элементов внутри родителя, не прибегая к созданию дополнительных модификаторов.

Вот наглядный пример использования такого подхода: в проекте используются стандартизированные стили кнопок, но в них не входит, например, задача внешних отступов. Ваша задача поместить кнопку в обозначенный блок и задать ей отступы. Можно, конечно, придумать ей дополнительный класс, скопировать в него большинство свойств и добавить еще внешние отступы. **Однако миксины позволяют упростить этот процесс разделив блок и элемент:**

```
<div class="search-form">
  <div class="button search-form__button"></div>
</div>
```

В рассмотренном примере класс button подгружает основные стили для кнопки, а класс search-from__button позволяет задать уникальные стили именно для кнопок, находящихся внутри блока search-from.

Правила оформления директорий

В небольших проектах ими можно пренебречь, но в крупных методологию БЭМ для построения иерархии файлов вполне можно использовать. **Сводится она к разбивки на три основные директории:**

- common – сюда вносятся файлы и папки, подходящие для большинства проектов;
- example – папка для примеров, но часто здесь выносятся html-файлы проекты, например, index.html;
- service – для разных специфических реализаций, применяемых только в этом конкретном проекте.

Вот пример структуры:

```
common/
 css/
 js/
 xml/
 xsl/
example/
 html/
service/
 auto/
  css/
  xml/
```

Правила работы с БЭМ в CSS

Благодаря наименованиями, разделенным по группам в CSS очень легко делать вложенности, особенно, если вы пользуетесь препроцессорами. **Вот пример стилей в БЭМ с применением препроцессоров:**

```
.block-name:
  стили блока
  &__element-name:
   стили вложенного элемента
  &_active:hover:
   стили элемента при наведении курсором
```

А вот пример реализации без использования препроцессоров:

```
.button_hovered .button__text
{
```

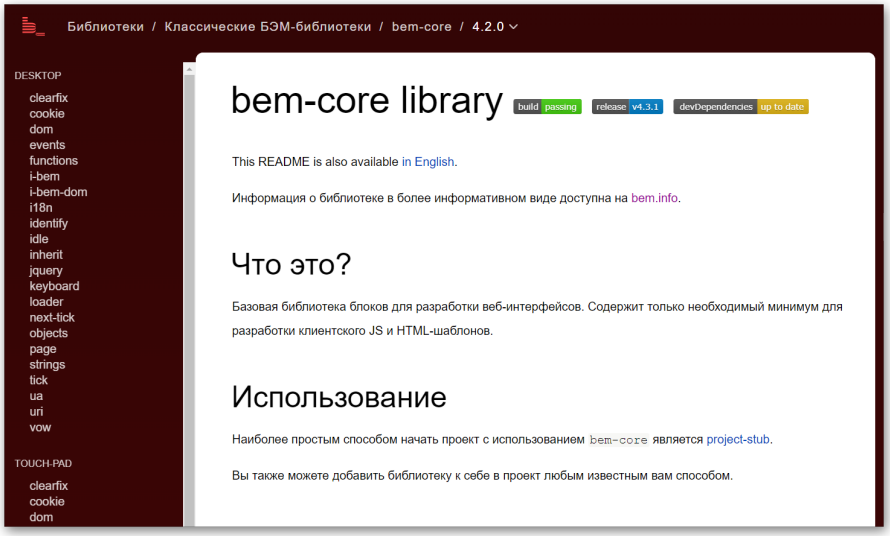
```
text-decoration: underline;
}

.button_theme_islands .button__text
{
line-height: 1.5;
}
```

БЭМ-библиотеки

Для упрощения работы с БЭМ можно найти в open source несколько библиотек, в том числе и официальных от Яндекс и неофициальных от сторонних разработчиков. **Вот две базовые:**

- bem-core – это библиотека с базовыми блоками, которая содержит JS-фреймворк i-bem и 20 блоков-помощников для ускорения разработки по методологии БЭМ.



Страница с документацией библиотеки bem-core

- bem-components – библиотека с готовыми визуальными компонентами, которые универсальны для большинства проектов. Внутри находится несколько контролов форм и базовых элементов для работы с интерфейсами.

Библиотеку bem-components можно подключать по аналогии с Bootstrap: **добавить предварительно собранные файлы библиотеки и вставить их в HTML страницы с помощью элементов link и script:**

<link rel="stylesheet" href="https://yastatic.net/bem-components/latest/desktop/bem-components.css">
<script src="https://yastatic.net/bem-components/latest/desktop/bem-components.js+bemhtml.js"></script>

Преимущества и недостатки БЭМ

Преимущества:

- Единообразие. Даже сделав беглый осмотр кода вы сможете понять, из каких блоков состоит страница, какие элементы в них вложены, как они будут себя вести на странице. Это значительно упрощает как разработку, так и поддержку готового проекта.
- Гибкость. Применение модификаторов, иерархической структуры позволяют быстро адаптировать проект под меняющиеся запросы, а также разным форматам устройств.
- Удобно работать в команде. Во-первых, вы и ваши коллеги отлично понимают структуру и последовательность, во-вторых, благодаря независимости блоков можно вести одновременную работу над разными элементами интерфейса.
- Простая оптимизация. Вместо того, чтобы лезть в код и искать место, которое нужно переписать, можно заменить отдельный блок. При этом можно не опасаться, что во время внесения изменений верстка развалится.

А вот недостатки:

- У HTML-элементов могут быть очень длинные названия, которые сложно записывать, однако легко запоминать, так как они выстраиваются по заранее определенной логике.
- Верстка по заданному в БЭМ шаблону не всегда удобна.

Про перспективы БЭМ

В среде разработчиков из СНГ БЭМ становится все более популярной – многие команды переходят на эту систему.

Даже начинающему верстальщику рекомендуется изучить эту методологию, чтобы успешно трудоустроиться.

На Западе популярны местные альтернативы: SMACSS и ECSS. **В целом, они похожи на БЭМ, но имеют некоторые особенности.** В-первом случае нет такого жесткого упора на названиях компонентов. Во-втором, упор делается на независимые компоненты. Тем не менее, отечественный БЭМ набирает популярность также и у западных разработчиков.

Заключение

Если вы хотите заниматься версткой и разработкой интерфейсов, то изучите методологию БЭМ хотя бы на базовом уровне. Ее применение сделает код более структурированным, легко поддерживаемым, позволит легче задавать стилистику проекта. **Несмотря на то, что вам придется давать компонентам сложные названия классов, сам процесс верстки сильно ускориться, да и вы будете меньше путаться в готовом коде.**

Добавить комментарий:

Имя *

E-mail *

Комментарий *

Перед публикацией комментарии проходят модерацию

ОТПРАВИТЬ

Новости и статьи

Обновления платформы

Смотреть новости

Документация по API insales

API для написания интеграций

Перейти на сайт документации

Разработка приложений

Пошаговая документация

Подробнее

Партнерская программа

Станьте партнером InSales

Подробнее о программе

Liquid

Переменные

Фильтры

Операторы и конструкции

API

Гайды

Github

Партнерская программа

Реферальная

Реселлерская

Шаблоны магазина

Разработка приложений

Конструктор лендингов

Магазин приложений

Лид-Центр

Сертификация

Университет

Технологии

Видео

Support

Контакты

Документация

Чаты, новости и обновления

InSales PRO

LiquidHUB